

ECE 344L Laboratory 0

Software Development Environment Familiarization

SPRING 2020

AUTHOR: DAVID KIRBY

DUE DATE: 06 FEBRUARY 2020

Laboratory 0

Software Development Environment Familiarization

Due Date: 6 Feb 2020

Name: David Kirby

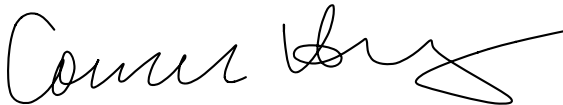
Points: 20 Points
Work individually.

Objective: The purpose of this laboratory is to familiarize you with the assembler/linker/loader system and its debugging capabilities. This lab will use the provided math.S routine, which will be assembled, downloaded, and executed. The MIPS instruction set information distributed in class will be useful to analyze what the program is doing.

Activities: Following the procedures given in the “How to use MPLab” that is provided with the assignment, generate a project using the math.S routine and execute it on the MX4 development platform. Add breakpoints so that you can examine the General Purpose Register before the execution of the loop and after the execution of the arithmetic loop. The program ends with a loop which merely “spins”; you will have to use the stop button to stop the program execution. Record the general purpose registers at the two points and include the results in your lab report. Similarly, examine the memory contents of where numbers_to_use are stored and the address specified by mem_lock and verify that the expected values are stored at those locations. **Demonstrate to the TA that you have found these values using the view memory and view register functions.** Experiment with the tools and build up a comfort level in using them. As part of this exercise, use the instruction summary sheet to determine what this program is doing and include an explanation in your report.

Documentation: Your lab activities must be documented following the guidelines that are provided on the course UNM Learn site. You must also demonstrate that your project functions properly to one of our TAs, who will then sign your copy of this assignment sheet. We have posted a schedule which indicates when they will be available in the lab to either help you, or witness your success. The TAs can also meet with you by appointment at other times – you can contact them to make arrangements.

Suggestion: Keep all of your files on a USB memory device as there is no guarantee that any information you store on lab machines will be preserved. On occasion, the machines must be cleaned and reloaded, so any information stored on them will be lost.



Introduction

The purpose of this laboratory was to familiarize students with the assembler/linker/loader system and its debugging capabilities. This lab used the provided `math.S` routine, which was then assembled and executed on the MX4 development platform. This laboratory was mission critical for this course as we will be using the MPLab software and the MX4 development platform extensively in our labs going forward.

Solution Methodology

Part I: `math.S` Routine

Following the procedures given in the “How to use MPLab” instructions provided on UNM Learn, we created a new project in MPLab version 8.92 using the Project Wizard. From there, we set up the software to run on PIC32MX460F512L hardware, compiled with the C32 C compiler, and added the provided `math.S` routine. With the routine built, we attached and programmed the chipKit Pro MX4 board and ran the instructions. Initially, the program performed the mathematical functions and went into an infinite loop (see Figure 1). Specified values are stored in `a` and `t` registers to be used for operations later in the code.

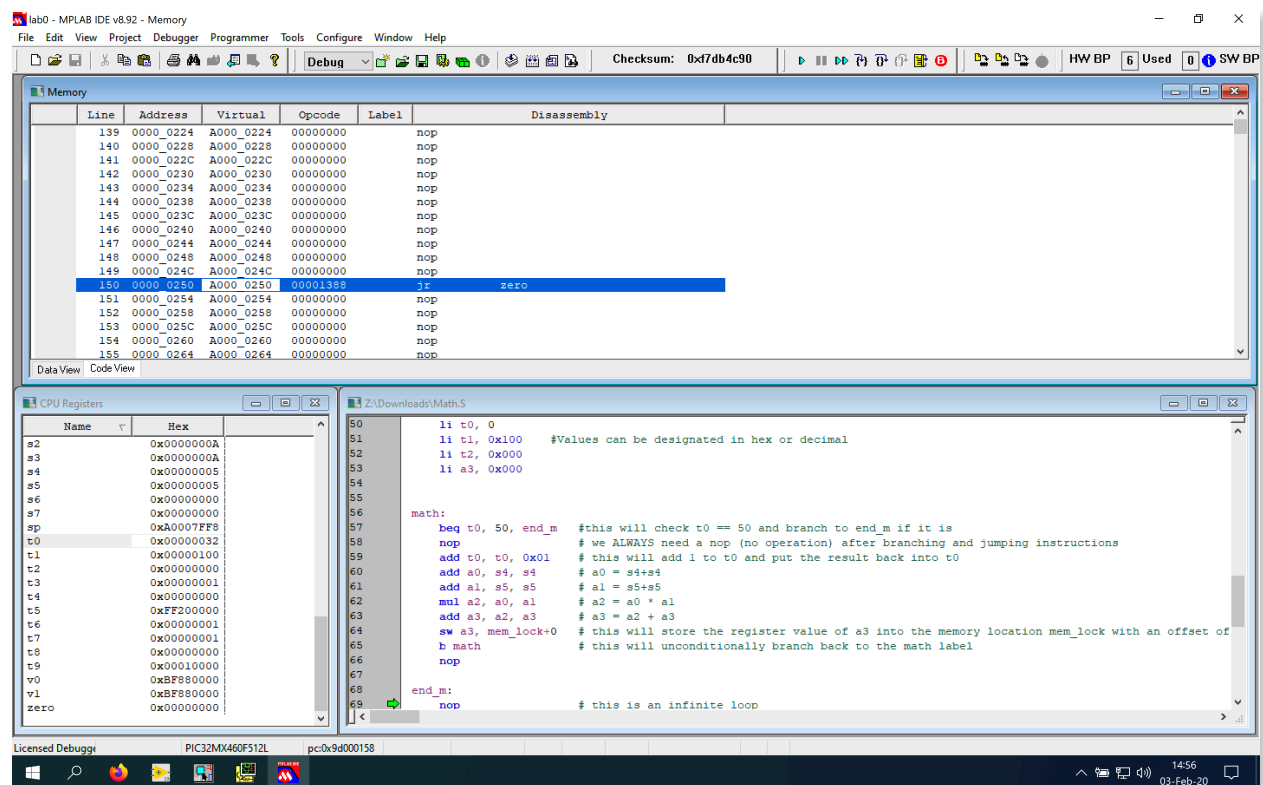


FIGURE 1

Part II: `math.S` Routine with Breakpoints

By either double-clicking on the side of the `math.S` routine or right-clicking and selecting *Add Breakpoint*, we were able to create breaks in the code which we could then analyze and determine the code behavior. We were tasked with creating breakpoints at specific portions of

the code and to examine the General Purpose Register before the execution of the loop and after the execution of the arithmetic loop (see Figure 2). The contents of the memory address A000_0250 is shown before and after math operations are performed. These operations are performed 50 times, as is evidenced by the t0 register changing from 0x00000000 to 0x00000032 - hexadecimal for 50. After the calculations were performed, the results in the memory were overwritten, eventually changing from 00000000 to 00001388.

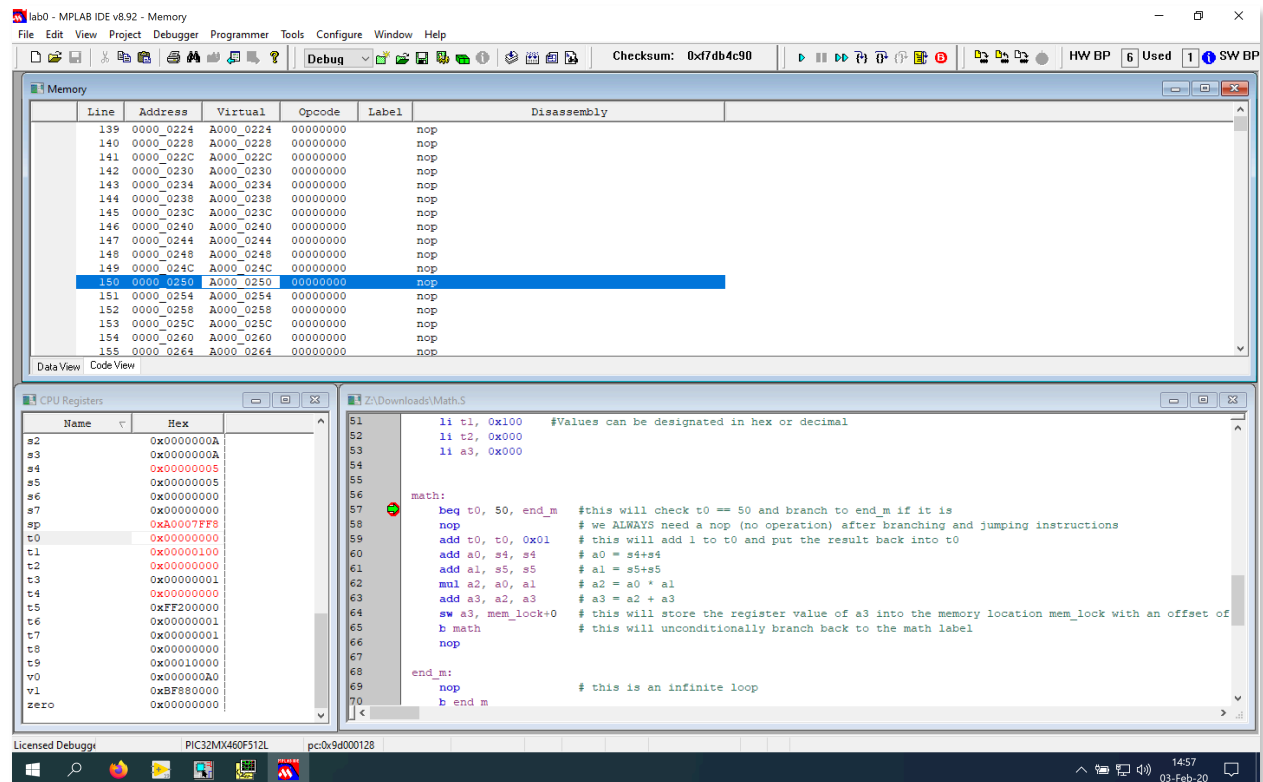


FIGURE 2

Source Code

No source code was generated, only code provided on UNM Learn.

Conclusion

Laboratory 0 was designed to familiarize students with the MPLab 8.92 development software and its debugger capabilities. It allowed students to jump into the provided code and to determine the registers and memory locations used to fetch, decode, and execute the instructions. Finally, it also allowed us to familiarize ourselves with Assembly code and to practice debugging.